

**Informatika Lietuvos mokyklose:  
žingsnis į priekį ar atgal?**

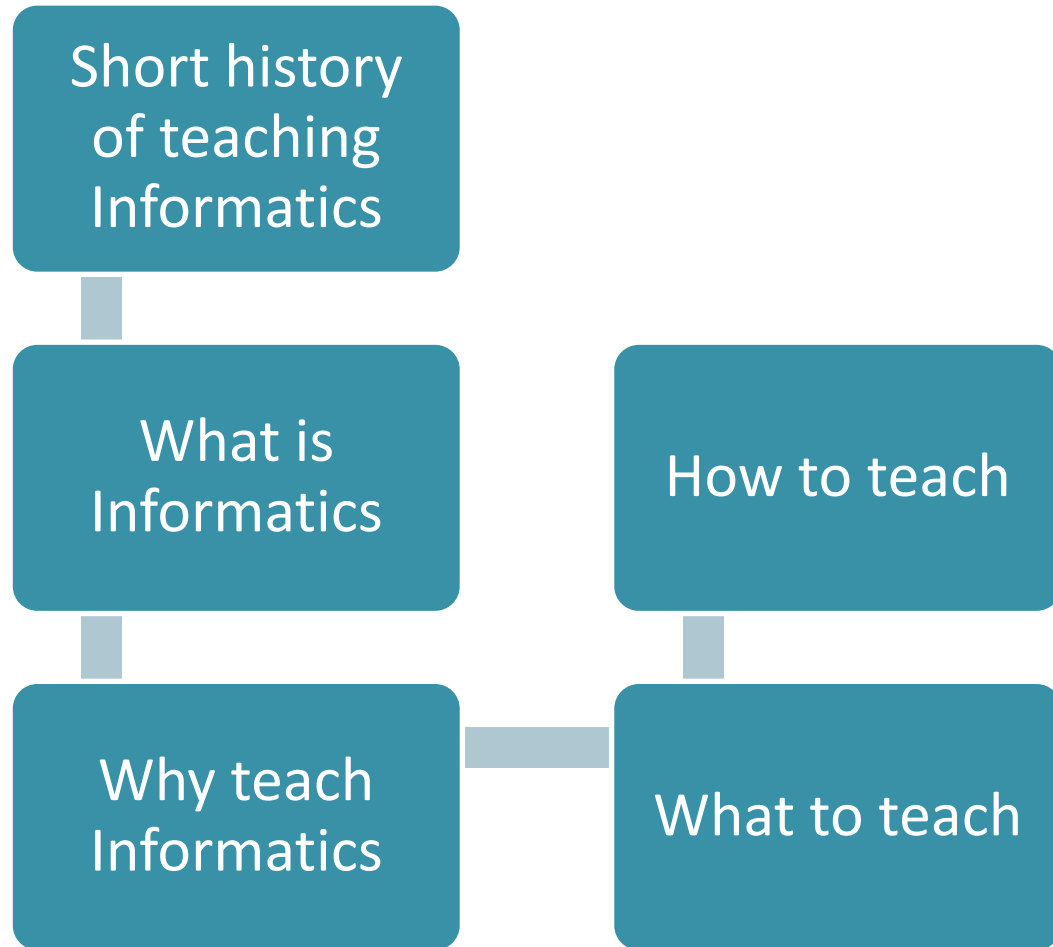
**Informatics in Lithuanian schools:  
step forward or back?**

Valentina Dagiienė

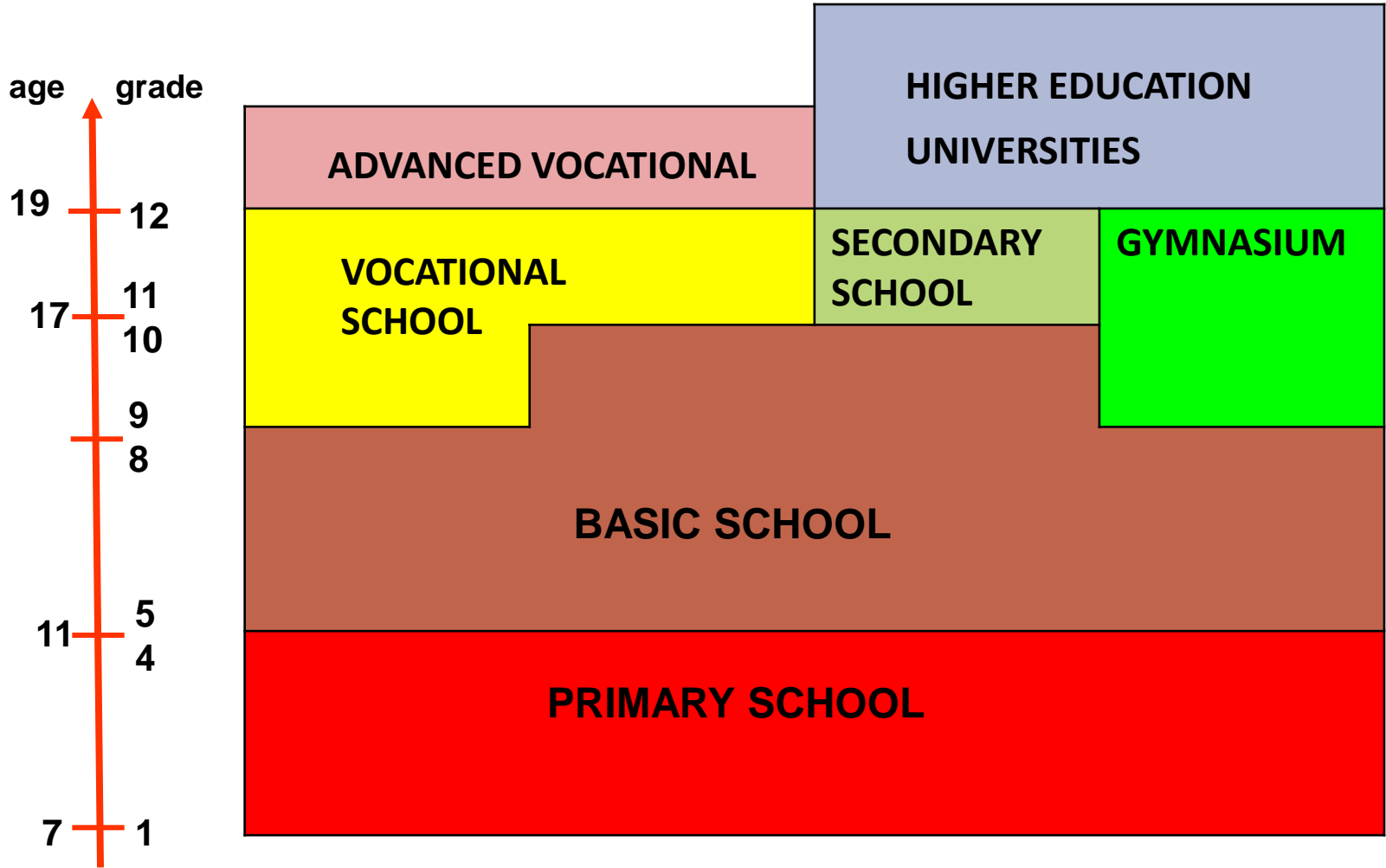
Matematikos ir informatikos institutas

Druskininkai, 2009 m. lapkričio 6-8 d.

# Overview



# The Structure of Education in Lithuania



# Early beginning, 1978-1985

## The Young Programmers' School by Correspondence

- **Established in 1981**
- **All students can participate**
- The activity of the Young Programmer's School in distance learning was one of the first examples concerning informatics and had a strong impact on many phenomena related with informatics' teaching
- **Learning by doing**
- **Problem solving**

The school continues to function nowadays - 29 years!

The first lessons of the Young Programmers' School published in newspaper "Komjaunimo tiesa" in 1981



### TRYLIKTOJI PAMOKA

Skyrelį tvarko LTSR MA Matematikos ir kibernetikos instituto jaunesnioji mokslinė bendradarbiė Valentina DAGEIENE

Programuotojai, rašantys neaiškias, grioždiskas programas, mėgsta teisintis, kad programa skiriama kompiuteriui, o ne žmogui. Be abejo, kompiuteriui programos aiškumas nesvarbus — jis mechaniskai atleka veiksmus ir nesidomi programos vaizdumu. Tačiau kad ir kaip atrodytų keista, didžiausias programų skaitytojas vis dėlto yra žmogus, o ne kompiuteris. Skaitydamas programas, žmogus susipažįsta su kitų programuotojų idėjomis ir patirtimi, mokosi pats sudarinėti programas. Dažnai tenka tobulinti ir pačių sukurtas programas. Visais tais atvejais reikia gilintis į programos esmę. Ką tik parašyta, dar šviežią atmintyje programą skaityti ne taip sunku. Tačiau ilgai neišėjus pamirštama. Todėl programos turi būti rašomos aiškiai, vaizdžiai, suprantamai.

Programavimo vadovėliuose vis daugiau dėmesio skiriama geram programavimo stiliui, sukurta nemaža taisyklių, kaip rašyti aiškias programas.

Ankstesnėse pamokose buvo kalbėta apie prasmingų vardų parinkimą. Tai kaip tik vienas iš daugelio programavimo kultūros elementų, gero stiliaus požymis. Prasmingai parinkti vardai leidžia greičiau suprasti programą. Tačiau pernelyg ilgi vardai nevertojami — užgriozdintų programą, o sutrumpinus dažnai pasidaro nebeaiški jų prasmė. Tada patogu įterpti į programą papildomą tekstą, kuris programos atlikimui neturėtų jokios įtakos, tačiau palengvintų ją skaityti. Toks tekstas vadinamas komentaris. Ko-

# PROGRAMAVIMO KULTŪRA

mentarais galima paaiškinti ne tik kintamųjų vardus, bet ir atskiras programas dalis, nurodyti, ką vienas ar kitas sakinytis atleka ir panašiai. Komentarus galima įterpti visur tarp atskirų simbolių, žodžių, skaičių, vardų. Jie suskaidžiami skliaustais (\*ir\*). Komentari padeda greitai ir lengvai skaityti programas. Tačiau jais nereikia piktnaudžiauti — komentari turi būti lakoniški, griežti, trumpai nusakantys pagrindinius dalykus, neužgriozdinantys programos teksto.

Paminėsime dar vieną programavimo kultūros elementą — programų redagavimą. Redagavimu vadinamas programos teksto išdėstymas popriečiau lape. Nekyla abejonių, kad žmogui kur kas lengviau skaityti vaizdžiai išdėstytą programą. Be to, tokioje programoje būna mažiau klaidų (pavyzdžiui, sunkiau pamiršti žodį end, jei jis rašomas po jį atitinkančiu žodžiu begin), lengviau jas surasti ir pataisyti.

Kaip kuo geriau suredaguoti programą, neretai priklauso nuo paties programuotojo — svarbu tik, kad būtų aišku ir vaizdu. Visose pamokose mes stengėmės pateikti suredaguotas programas. Laikėmės kai kurių bendrų redagavimo taisyklių: sakinius, esančius kitame sakinyje, patraukdavome į dešinę per keletą pozicijų, vertikalčiai lygiavome žodžius begin ir end ir panašiai.

Pavyzdys. Sudarysime programą populiariam matematikos uždaviniui, kurį 1202 metais suformulavo Italų matematikas Fibonačis.

Triušių pora kas mėnesį atsiveda du triušius (patelę ir patinėį), o iš atvestųjų triušiu po dviejų mėnesių jau gaunamas naujas prieauglis. Kiek triušiu bus po metų, jei pradžioje turėjome vieną subrendusią triušių porą?

Iš sąlygos matyti, kad pirmojo mėnesio pabaigoje turėsime dvi triušių poras. Ant-

rojo mėnesio pabaigoje prieauglį duos tik pirmoji pora, todėl turėsime tris poras, o dar po mėnesio prieauglį duos ir pradinė pora, ir pora, gimusi prieš du mėnesius. Todėl iš viso bus 5 poros.

Simboliu F(n) pažymėkime triušių porų skaičių, kurį turėsime po n mėnesių. Matome, kad n-ojo mėnesio pabaigoje turėsime tiek porų, kiek jų buvo prieš mėnesį, t.y. F(n-1) ir dar tiek naujų porų, kiek jų buvo prieš du mėnesius, t.y. (n-2)-ojo mėnesio pabaigoje. Kitaip sakant, gausime tokią priklausomybę:

$$F(n) = F(n-1) + F(n-2)$$

Patelksime programą triušių porų skaičiui po n mėnesių spausdinti.

```
program fibonacii;
var fn, (* F(n) *)
    fn1, (* F(n-1) *)
    fn2, (* F(n-2) *)
    n, (* mėnesių skaičius *)
    k: integer;
```

```
begin
fn1:=1; (* F(-1) *)
fn:=1; (* F(0) *)
read(n);
for k:=1 to n do
begin
fn2:=fn1; (* praėjo *)
fn1:=fn; (* vienas *)
fn:=fn1+fn2>(*mėnuo*)
end;
write(fn)
end.
```

Kai pradinis duomuo 12, kompiuteris išspausdins skaičių 377. Vadinasi, po 12 mėnesių turėsime 377 poras triušių.

### KONTROLINIŲ UZDAVINIŲ SPRENDIMAI

10. U===U  
U===U  
U===U  
UUUU

11. Praleistos sąlygos  $d \leq x$  ir  $x \times x \bmod d = x$ .

12. Vienas iš paprasčiausių sprendimų būtų šitoks: program dalumas; var n, s, d: integer; begin read(n);

```
for s:=1 to n do
begin
write(s);
for d:=1 to n do
if s mod d=0 then
write(' ');
writeln
end
end.
```

Jis nėra efektyvus, tačiau kadangi iš uždavinio sąlygos nereikalau, kad n būtų labai didelis, tai skaičiavimų bus ne daug ir nėra reikalo šiuo aspektu tobulinti programos.

### KONTROLINIAI UZDAVINIAI

13. Duota programa: program atspėk; var a, b, c, i, j: integer; begin read(n); a:=1; b:=1; for i:=0 to n do begin for j:=1 to b do write('\*'); writeln; c:=a+b; a:=b; b:=c end end.

Ką išspausdins kompiuteris, atlikęs šią programą, jei pradinis duomuo yra 7? Kokio uždavinio sprendimas užrašytas šia programa? (7 balai)

14. Pradiniai duomenys — natūrinių skaičių seka. Sekos pabaigos požymis — nullis. Sudarykite programą mažiausia ir didžiausia sekos nariui spausdinti. Seką užbaigiantis nullis sekos nartu nelai-komas. (9 balai)

15. Pradiniai duomenys — du natūriniai skaičiai a ir b reiškiantys stačiakampio kraštinių ilgius. Sudarykite programą, kuri suskaičiuotų, kiek mažiausiai kvadratų, kurių kraštinės išreiškiamos natūriniais skaičiais, galima padalyti duotąjį stačiakampį. Pavyzdžiui, kai pradiniai duomenys 5 ir 2, tai rezultatas turi būti 4. (10 balų)

Sprendimus išsiųskite ne vėliau kaip š. m. gruodžio 17 d. Adresas: 232021 Vilnius, Akademijos g. 4, Matematikos ir kibernetikos institutas, Jaunųjų programuotojų mokykla.

Ši pamoka paskutinė — baigėme programavimo pradmenų kursą. Apie mokyklos darbo rezultatus, tolmesnius planus parašysime vėliau.

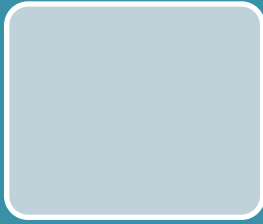


# Obligatory course of Informatics in 1986

- The official beginning of informatics as a subject in Lithuanian schools: 1986
- A founder of the Siberian School of Computer Science prof. Andrei Ershov announced: “Programming is the second literacy“
- Idea: to introduce every student with computers through programming



# Informatics content in 1986



## Algorithms

- Formal language (special for algorithms)
- Variables, loops, arrays...



## Technical issues

- Hardware
- Information processing

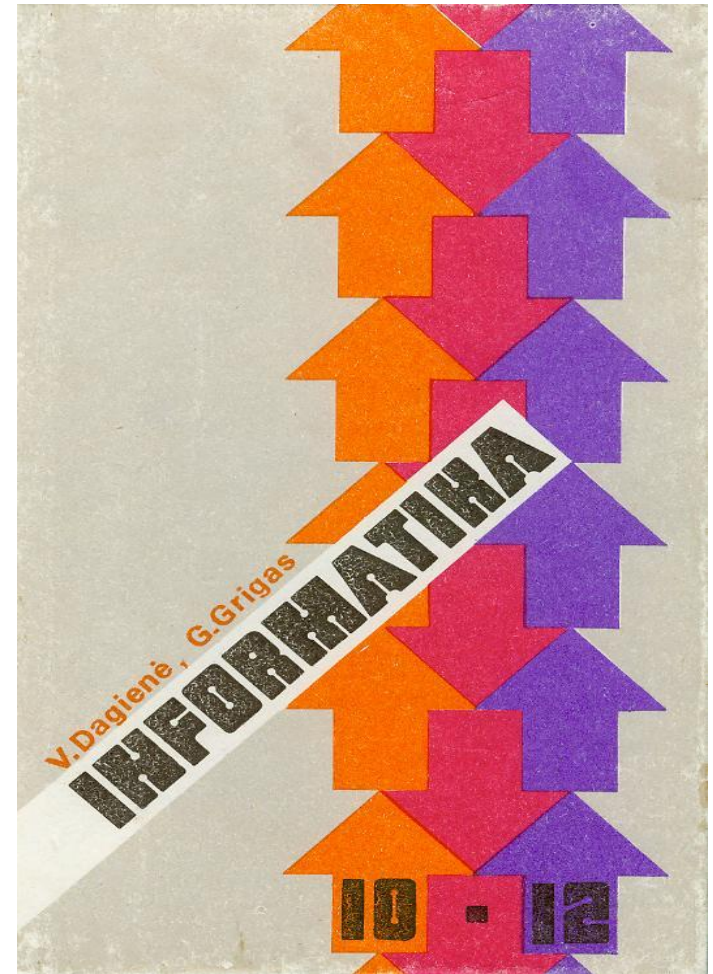


## Programming

- Basic
- Rapyra
- Pascal

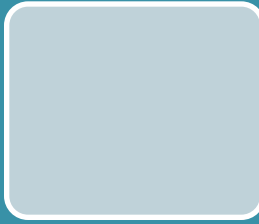
# Obligatory course of Informatics in 1990

- An original Lithuanian textbook of Informatics was published just after Lithuania has regained independence
- In parallel, in 1991, the first curriculum for teaching informatics in secondary schools was developed





# Informatics content in 1990



## Algorithms

- Pascal-based
- Data types
- Variables, loops...
- Task solving based



## Information

- Coding and representation
- Information measuring



## Logics

- Operations
- Laws
- Schemes

# Informatics content in 1999: towards IT

## Obligatory course

- Text processing
- Working with the spreadsheet
- Presentation
- **Cognitive, social, and ethical issues**
- Surfacing of WWW and communicating

## Optional modules

- **Programming**
- Data base
- Multimedia

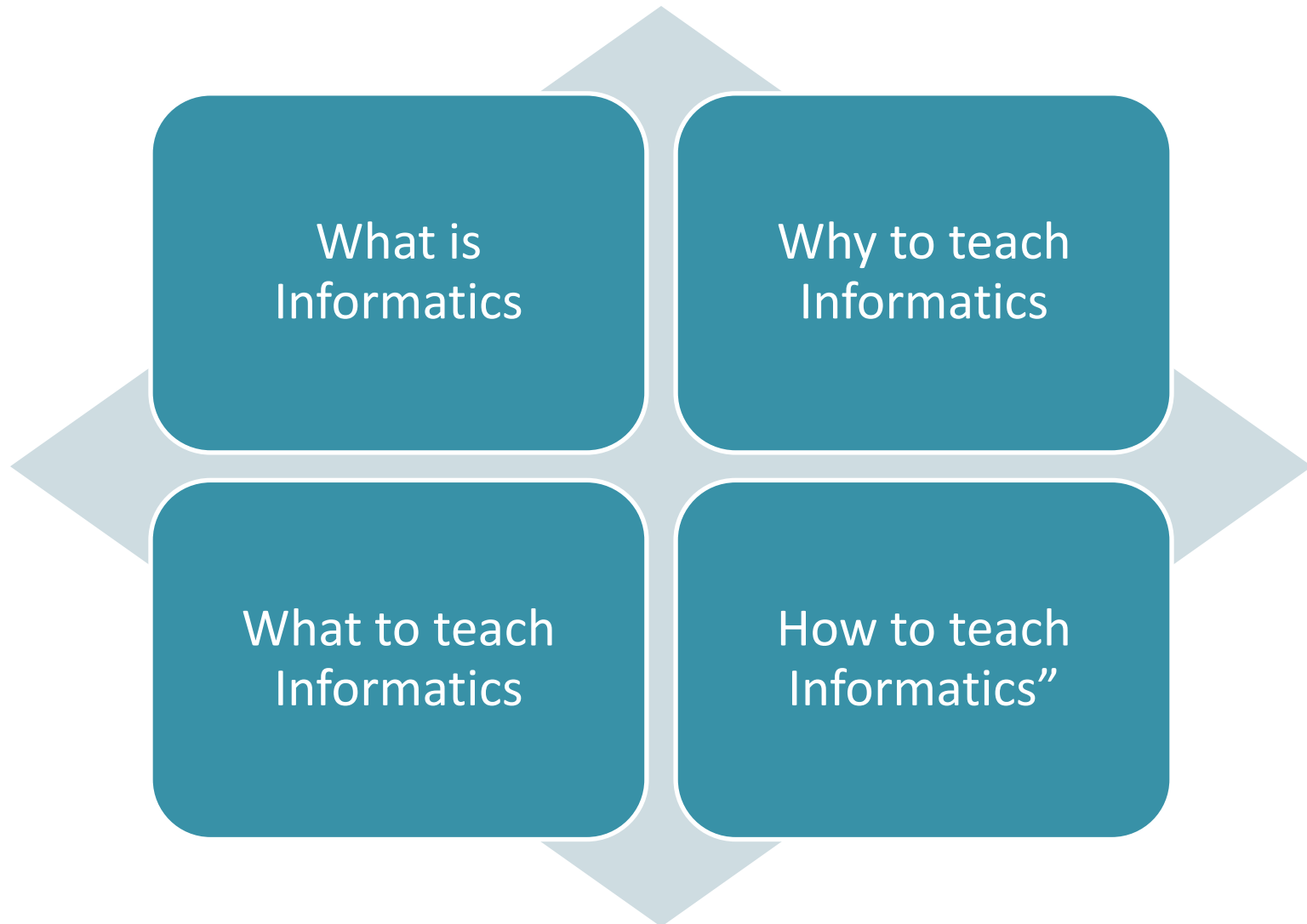
# Teaching Informatics / IT in 1986-2009

Years	Grades	Mandatory	Optional
1986-1997	10-12	<b>Informatics</b> 2 lessons / week	Various IT modules
1997-2004	Basic school, grades 9-10	<b>Informatics / IT</b> 2 lessons / week	
	Secondary school, grades 11-12	<b>Informatics / IT</b> 2 lessons / week	Modules 2-4 / week
Since 2004	grades 5-6	<b>Informatics / IT</b> 2 lessons / week	
	grades 7-8	IT, 1 lesson / week 1 integrated / week	
	grades 9-10	IT 1 lessons / week	<b>Programming</b> module
	grades 11-12		IT / <b>Programming</b> modules

# Some things change, some stay the same...

	<b>Then (mostly)</b>	<b>Now (extending to ...)</b>
<b>Students</b>	<b>digital immigrants</b>	<b>digital natives</b>
<b>Technology</b>	<b>“The Internet”</b>	<b>e- and m-Learning</b>
	<b>asynchronous text</b>	<b>synchronous voice, video</b>
	<b>broadcast web sites</b>	<b>social software</b>
<b>Theory</b>	<b>constructivism</b>	<b>connectivism!!!</b>
<b>Teachers</b>	<b>digital immigrants</b>	
<b>Contexts</b>	<b>distance and blended “classrooms”, all levels, many industries</b>	
<b>Research focus</b>	<b>efficacy of technology in specific classroom/context, student as learner and teacher activities</b>	
<b>Reflection and change</b>	<b>less reflection on, or take-up of, higher level lessons, e.g., new theories and models, sustainability</b>	

# Teaching (and Learning) Informatics

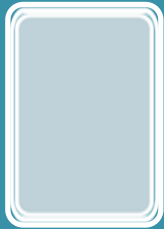




# What is Informatics / Computer Science?



Informatics is defined as the science dealing with *design, realization, evaluation, use, and maintenance of information processing systems*, including hardware, software, organizational and human aspects (by UNESCO)



Information Technology is defined as *the technological applications of informatics in society* (by UNESCO)

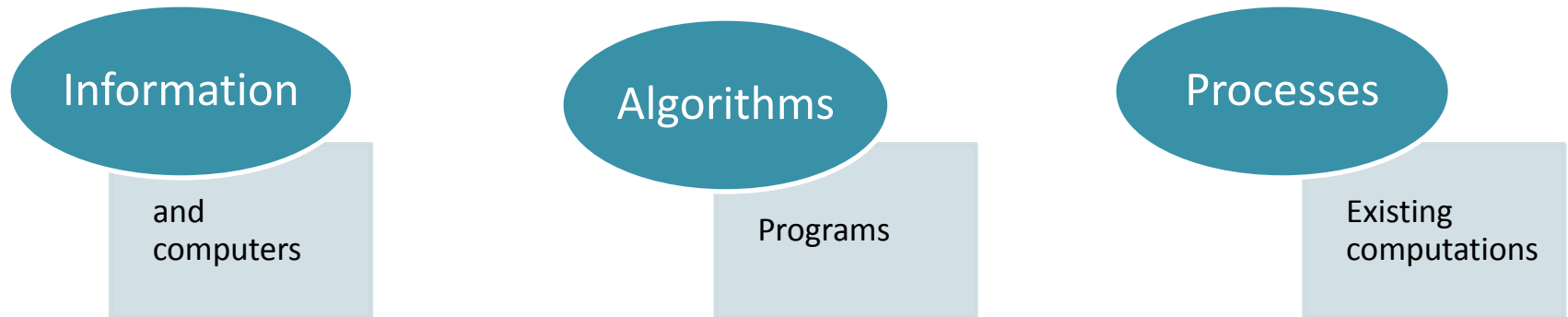


Informatics is the science of *algorithmic processing, representation, storage and transmission of information* (by J. Hromkovic)

# To which scientific discipline does **Informatics** belong?

Informatics investigates **general categories**: information, algorithm, randomness, complexity, language, knowledge, communication, simulation, determinism, approximation, etc.

# The objects of investigation of Informatics



# Why to teach Informatics?

Informatics is interdisciplinary: it is focused on the search for solutions for problems in all areas of sciences and in every day life.

Informatics –  
fascinated  
scientific  
discipline

Informatics has made great contributions to our view of the world – through its spectacular results and high interdisciplinarity.

# Why to teach Informatics?

## Philosophical depth

Are there problems that are not algorithmically solvable?

How does one define the difficulty (hardness) of problems?

## Applicability results

Informatics provides concepts and methods that can be applied during the whole process of design.

It can take part in many frontiers of research: medical diagnostics, speech recognition, space exploration, etc.


## Way of thinking

Informatics encourages creating and analyzing mathematical models of real systems

It is powerful enough to attack complex real-world problems.



# What to teach



Learning **programming** means learning a language of communication with technical systems.

Teaching fundamentals could start with **automata theory**: finite automata provides the simplest model of computation.

Computability?

Everything is matter of didactic mastery

(J. Hromkovic. Sieben Wunder der Informatik: Eine Spaziergang an der Grenze des Machbaren, 2006)

(Algorithmic adventures: Moving the limits of doable, 2006)

# How to teach Informatics

## Support of Iterative Teaching

- The core of textbooks is the formalization of informal ideas by theoretical concepts and to the study within the framework of these concepts.


## Less is Sometimes More

- Dedicate more time to the motivation, aims, connection between practice and theoretical concepts, and especially to the internal context of the presented theory.

## Simplicity and Transparency

- Clarity takes priority over the presentation of the best known results.

# Future investigations...

- 
- Search for means to measure and compare contributions of the different teaching concepts in informatics.
  - Define standards for executing and evaluating experimental teaching in informatics.
  - Intensive work on teaching materials, textbooks, e-learning systems and teaching environments.



It is time to start enforcing informatics didactics as a serious discipline.